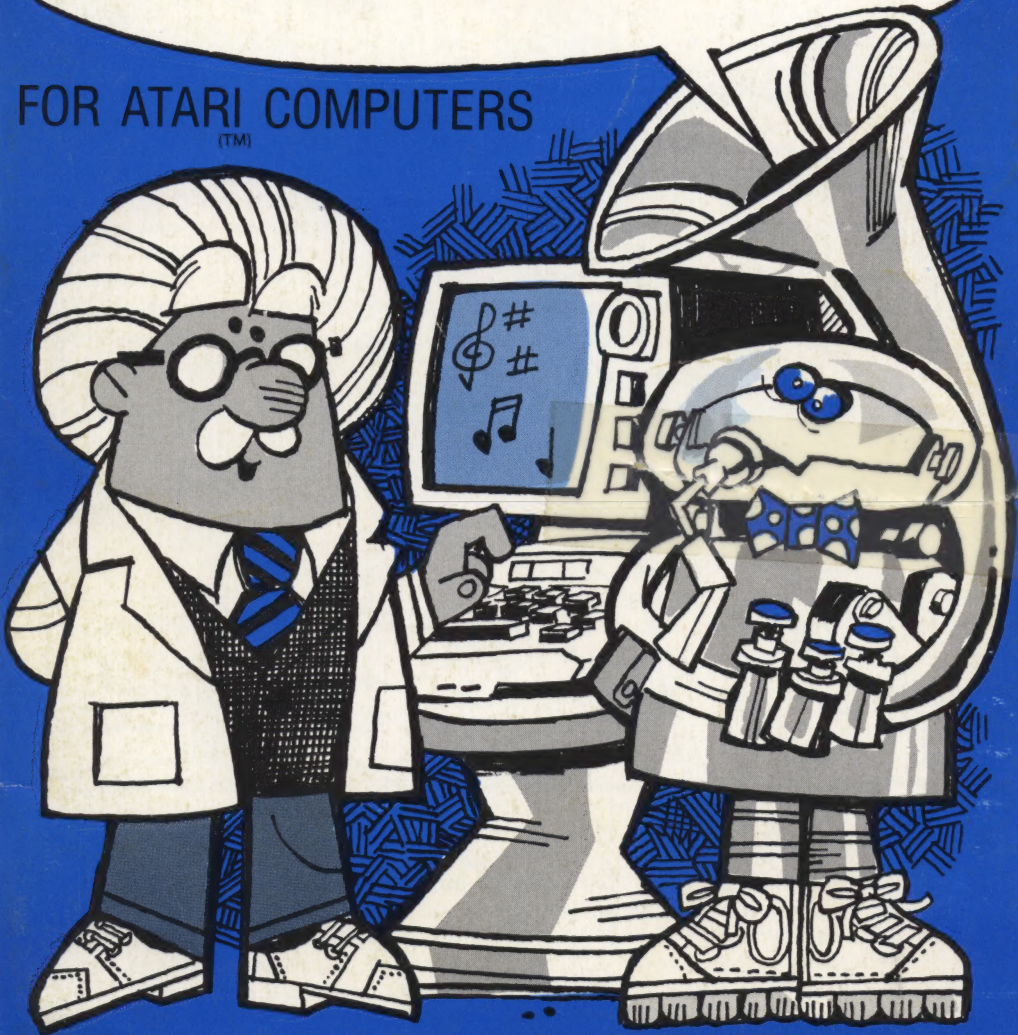
SOUND & MUSIC

TRICKY TUTORIAL #6 (TM)

INCLUDING PLAYER PIANO

BY JERRY WHITE

FOR ATARI COMPUTERS (TM)



WE MAKE LEARNING TO PROGRAM FUN

Educational Software inc.
SOQUEL CA

ALSO FROM EDUCATIONAL SOFTWARE

Professor Von Chip's TRICKY TUTORIALS

Here, at last, are the means for knowing, using, and understanding the great power inherent in the ATARI 400/800. Von Chip's TRICKY TUTORIALS take the mystery out of programming by offering several coherent lessons such as:

Tricky Tutorial # 1 -- DISPLAY LISTS
A lesson on mixed graphic screens -- $19.95

Tricky Tutorial # 2 -- HORIZONTAL/VERTICAL SCROLLING
Scroll up, down, sideways, or even diagonally -- $19.95

Tricky Tutorial # 3 -- PAGE FLIPPING
Change screens instantly! -- $19.95

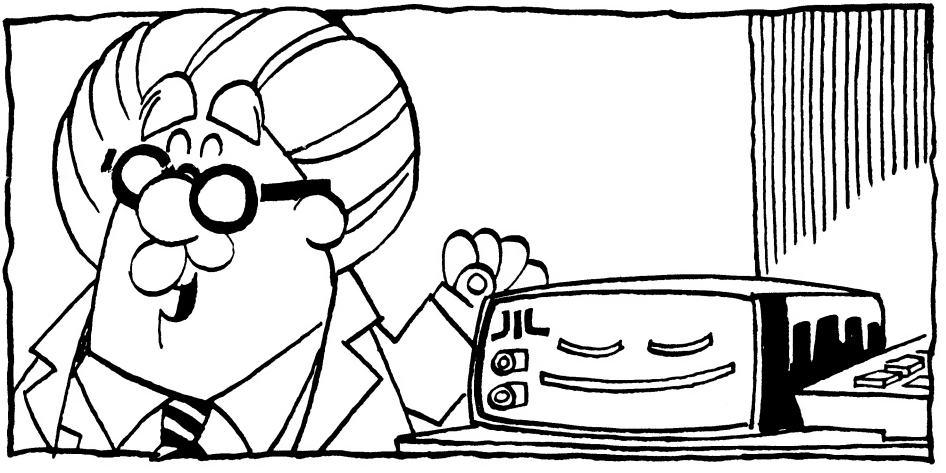
Tricky Tutorial # 4 -- BASICS OF ANIMATION
Beginning animation the easy way. -- $19.95

Tricky Tutorial # 5 -- PLAYER MISSILE GRAPHICS
Create your own video games and animate them. -- $29.95

Tricky Tutorial # 6 -- SOUND and MUSIC
Program your own music, includes PLAYER PIANO -- $19.95

The 1st Six Tricky Tutorials -- Discount Price -- $99.95

Tricky Tutorial # 7 -- DISK UTILITIES
Von Chip's tricks on using disk drives. -- $29.95



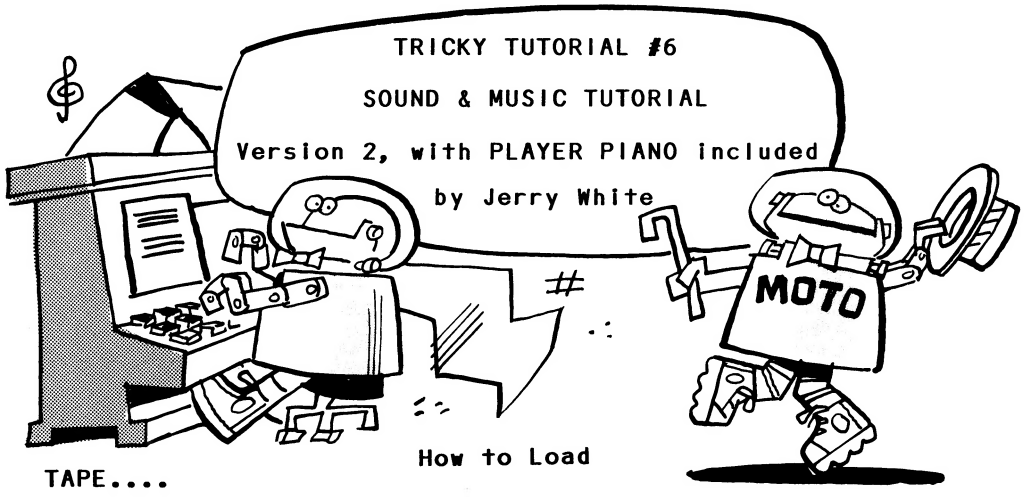
Coming Soon from EDUCATIONAL SOFTWARE

Tricky Tutorial # 8 -- CHARACTER GRAPHICS
Make your own character sets and animate them.

Tricky Tutorial # 9 -- GTIA, GRAPHICS 9 TO 11
16 Colors at once!

Tricky Tutorial # 10 -- SOUND EFFECTS
Find that perfect sound effect for your program.

Tricky Tutorial # 11 -- THE MEMORY MAP TUTORIAL
30 lessons demonstrating useful memory locations.



How to Load

TAPE....

Place the tape in your recorder, label side up. Make sure the tape is rewound, and reset the counter to zero. Push PLAY on the recorder, type CLOAD and press RETURN. If the program won't start to load, try positioning the tape forward or backwards a little. A little trick to find the beginning is to first, turn your volume UP. Then POKE 54018,52 to start the cassette motor. Listen to the "noise" on the tape. When you find the high-pitched, steady tone, you have the beginning of the program. We recommend you write down the number on your recorder's counter as each program starts, this will make it easier to find each part later on. POKE 54018,60 to turn the cassette motor off.

DISK....

To load and run the disk, first turn on your disk drive. When the busy light goes out, place the disk in the drive. Now turn on the computer, with the BASIC Cartridge in place and the program will load each part and run by itself.

Requirements:

Atari Basic Cartridge
ATARI 810 Disk Drive (with 24K of memory) or
ATARI 410 Program Recorder (with 16k)

Optional:

Any ATARI or equivalent Printer

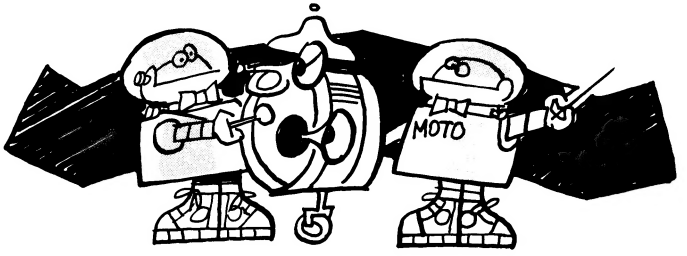
24K of memory (for Tape) or 32K of memory (for Disk) required to run the bonus PLAYER PIANO program

- CONTENTS -

INTRODUCTION

This software package was designed to provide ATARI BASIC demonstrations of your ATARI computer's vast SOUND capabilities. This program starts with the simple SOUND statement, but progresses to chords and complete songs. Several song programs play music in four part harmony while they display lyrics on the screen. Although challenging, this tutorial, combined with your own creative skills and determination, provide you with all the information neccessary to become a virtuoso ATARIST.

If you have a minimal understanding of ATARI BASIC commands and at least some understanding of music in general, this Tutorial will provide a useful introduction to sound production without an extensive programming background. These programs were designed to aid someone interested in learning how to create music and various sounds using ATARI BASIC. If you just want to play music, a program is included that allows you to play the keyboard like a piano and SAVE the music for future listening. Thus all ages will find something they can enjoy in this lesson!



***** How to Use This TRICKY TUTORIAL *****

This tutorial begins by teaching the basic parameters of the SOUND statement and the dynamics of chord structure. It then progresses to complete songs. All of the material can be used by a beginner, but an understanding of BASIC would be helpful, especially if you plan to make modifications. You will need some knowledge of music to write your own songs, however this doesn't mean you can't have some fun with little tunes or sound effects (watch for our upcoming tutorial on Sound Effects). If you don't understand the following musical terms, you may want to look them up or see our new Fundamentals of Music lesson called MUSIC MAJOR coming soon from Educational Software.

NOTE or PITCH, CHORD, SHARP, and FLAT

Look at all of the programs first. Then, when you find a single topic you want to work with or understand better, LOAD that program into your memory and LIST it to your screen. Tape users should write down the starting number of each program on their tape. This will make it easier to find a desired program later.

Look at the program LISTings to see how they were written. Remember, at any point while you're RUNning a program you can just hit the BREAK key, type GRAPHICS 0 to clear the screen, and LIST. Think of your own ways to modify this TRICKY TUTORIAL and go at it. I suggest you make a copy and modify the copy, keeping your TRICKY TUTORIAL disk or tape as the master. You can write protect master disks by covering the small slot in the edge with a piece of opaque tape.

Here are two methods I'd like to explain that may help in case you get confused while trying to read the program code. First, try acting as if you are the computer. Some guy named BASIC is your boss and he is giving you detailed instructions. You take each instruction, one at a time, and do exactly as you are told. The computer probably has a better memory than you have, so keep a pencil and paper handy. The instructions that you (the computer) execute will change things. Write these things down and make changes as needed.

For example, if the numeric variable X were equal to 5, you should have "X=5" written down. Then if BASIC told you that X=X+1, draw a line through "X=5" and write down "X=6".

Second, you might also BREAK into the program as it is RUNning, then make changes. Then reRUN the program or routine, and see how your changes affected the program. If it stops running you'll know why I asked you to make a backup copy of the master.

If you BREAK into a program while the SOUNDs are on, you may wish to shut them off. Three easy ways are 1) Type in "END" and press RETURN, 2) press the SYSTEM RESET key, or 3) turn off the volume on your TV or monitor. The latter will not affect the SOUND channels in the computer. The computer doesn't even care if you turn off your TV completely, the computer will still be running!

It's up to you to experiment with sound. This TUTORIAL is the foundation. Write and let us know what you think of it.

Good Luck!

Professor Von Chip

Professor Von Chip and Friends

THE SOUND COMMAND
Screens 1 to 8

The SOUND command must be followed by four parameters for the ATARI to understand what you're talking about. These parameters can be numeric constants (like 2, 14, etc) or numeric variables (A, Z-10, etc). This is a sample SOUND command:

SOUND VOICE,PITCH,DISTORTION,VOLUME

For Example:

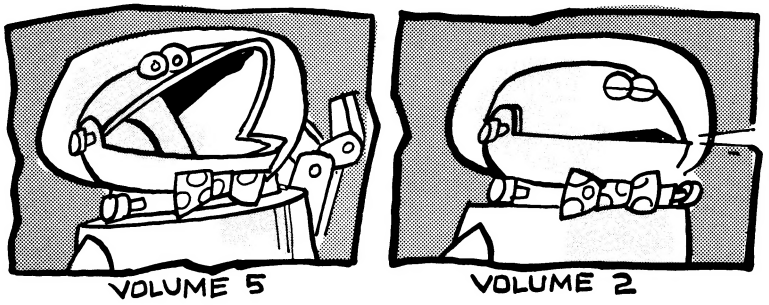
SOUND 0,100,10,8

The first parameter is called the VOICE. Your ATARI computer has four sound channels, or voices, numbered zero through three (have you ever noticed nothing ever seems to start with a 1 anymore?).

The second parameter is called PITCH. The pitch can be any integer value from 0 through 255. When the clear sound of distortion level 10 is used, increasing the PITCH value makes the sound deeper or lower, and decreasing the PITCH value makes the sound higher. And all without hormones!

The third parameter is DISTORTION. This can be any even number value from 0 through 14. A clear, or undistorted, sound is obtained by using a value of 10 or 14.

The fourth parameter is VOLUME. This may be any integer value from 0 through 15. A value of zero shuts off a sound. (don't you wish you could poke a zero into some people and shut off their sound?). As the value increases, the sound becomes louder.



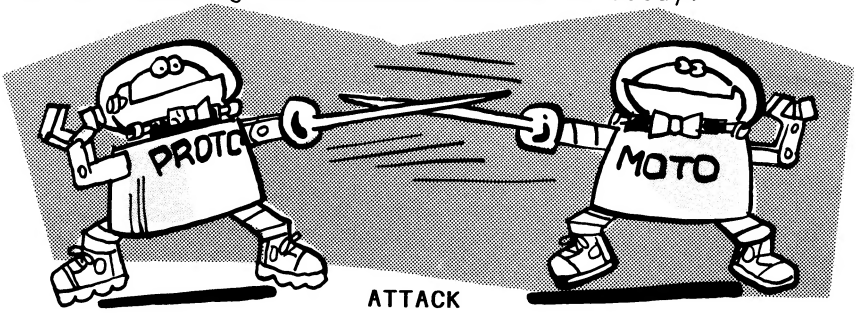
As the program runs, it provides it's own instructions on the screen. Your input will be requested from time to time, just make sure you are not using lower case or inverse video in your replies. That is, if you want it to work.

The variables used in the program are P=Pitch, D=Distortion, and V=Volume.

ATTACK AND DECAY

Screen 6

These two parameters of music are demonstrated by screen 6 within the Sound Command program, but I felt that some additional examples would help. Have you ever wondered why middle "C" sounds different on various musical instruments? While the reasons are quite complicated, a major factor is the attack and decay of a note. Attack refers to how long it takes for a note to reach its maximum volume. The speed with which the loudness of a note falls off after reaching its maximum volume is decay.



ATTACK

In ATARI BASIC, you can create a note with a controllable attack by using a FOR NEXT loop with the loudness parameter of a SOUND statement as the variable. Here is an example you may type in to experiment with the attack of a note (middle C):

```
100 REM --attack loop
110 FOR L=0 TO 15
120 SOUND 0,121,10,L
150 NEXT L
155 SOUND 0,121,10,0:REM TURN SOUND OFF
```

Since BASIC will increase the volume so fast you can't hear it,we can lengthen the time of the attack by putting a delaying pause into our program. In our example, we added two lines of code so our program now reads:

```
100 REM--attack loop
110 FOR L=0 TO 15
120 SOUND 0,121,10,L
130 FOR P=1 TO 10
140 NEXT P
150 NEXT L
155 SOUND 0,121,10,0
```

This effectively slows the attack of our note for an audible difference. Feel free to experiment by changing the length of the pause loop. The longer the loop (a larger value of P), the slower the attack.

DECAY

To affect the decay of a note, we must set up a loop and count the loudness parameter in reverse using the STEP command.

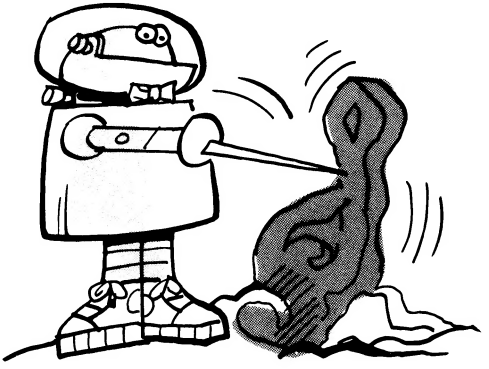
```
160 REM--decay loop
170 For L=15 TO 0 STEP -1
180 SOUND 0,121,10,L
190 FOR P=1 TO 10
200 NEXT P
210 NEXT L
```

Like our attack example, we can extend the delay period by adding a small delay loop (lines 190 and 200) which will cause the sound to be drawn out as its volume decreases to 0. Notice that the sound statement to return the volume to 0 is no longer needed. Listen carefully to screen 6, or type in these examples to hear the differences.

Using these techniques, we can simulate the sound of a piano by giving the note a fast attack or we can simulate a violin by giving the note a slow attack. A longer decay makes our note sound more like a harpsichord.

ATTACK AND DECAY TOGETHER

The back-to-back loops used with our previous examples can be incorporated into an example of attack and decay on the same note. Try the following example to hear them used together. Most of the sounds you will hear in the remaining screens use both attack and decay to control their sound. This is done by combining the above lines:



```
100 REM--attack loop
110 FOR L=0 TO 15
120 SOUND 0,121,10,L
130 FOR P=1 TO 10
140 NEXT P
150 NEXT L
160 REM--decay loop
170 FOR L=15 TO 0 STEP-1
180 SOUND 0,121,10,L
190 FOR P=1 TO 10
200 NEXT P
210 NEXT L
```

To hear the note repeat add:

```
220 GOTO 100
```

By changing the number 10 in lines 130 and 190, you can experiment with the attack and decay of a note (use a larger or smaller number).

If you were to BREAK screen number 6, you would see all of the above ideas incorporated into several BASIC statements numbered 920 to 980. The parameters are the same, just more compact to run faster.

SOUNDEXP
Screen 9

Use this nice little program to become more familiar with the SOUND statement. It also may be considered an Editor to use when trying to get that special sound you need. Since I have placed the instructions on the screen, I will not bore you with it all again. What? You're bored anyway. Well, we are finally getting to the good part, Basic Music:

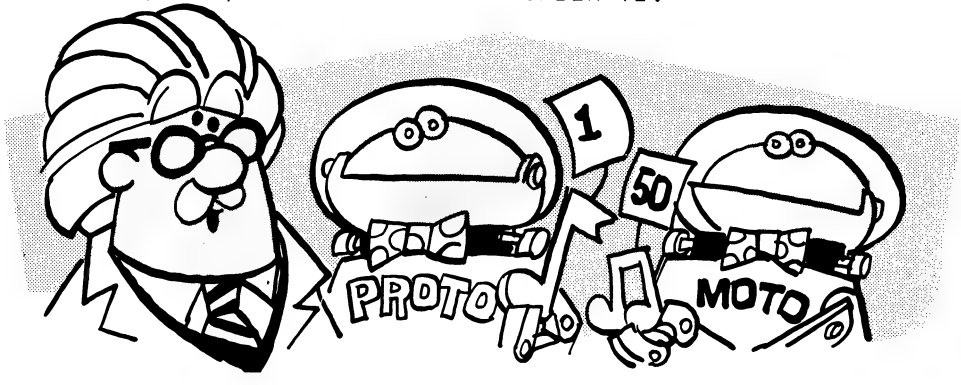
BASMUSIC
Screens 10 to 17

This example begins by referring you to a pitch chart that will become the most looked at page of this TUTORAL. Here, then, is :

THE PITCH CHART
FOR
DISTORTION LEVEL 10

NOTE #	PITCH	MUSICAL NOTE
1	14	C
2	15	B
3	16	A# or Bb
4	17	A
5	18	G# or Ab
6	19	G
7	21	F# or Gb
8	22	F
9	23	E
10	24	D# or Eb
11	26	D
12	27	C# or Db
13	29	C
14	31	B
15	33	A# or Bb
16	35	A
17	37	G# or Ab
18	40	G
19	42	F# or Gb
20	45	F
21	47	E
22	50	D# or Eb
23	53	D
24	57	C# or Db
25	60	C
26	64	B
27	68	A# or Bb
28	72	A
29	76	G# or Ab
30	81	G
31	85	F# or Gb
32	91	F
33	96	E
34	102	D# or Eb
35	108	D
36	114	C# or Db
37	121	C
38	128	B
39	136	A# or Bb
40	144	A
41	153	G# or Ab
42	162	G
43	173	F# or Gb
44	182	F
45	193	E
46	204	D# or Eb
47	217	D
48	230	C# or Db
49	243	C
50	255	B

The normal notes in the musical scale that can be reproduced on the ATARI are shown. As the screen explains, my pals PROTOTYPE and MOTOTYPE have finally come up with something useful!!! (I did offer a little of my Professorial help). We have numbered the notes in the chart from 1 to 50. The program saves the corresponding pitch values in an array called N(P). If you don't know what an array is, just look in your Basic manual, or ignore it. You can use this method without knowing why it works. I'm not trying to avoid answering your questions, it's just that we are not trying to teach Basic programming in this TUTORIAL. Of course, you could always buy TRICKY TUTORIAL NUMBER 12.



The next screen, number 11, plays all 50 notes in the array using this BASIC statement:

```
310  FOR P=1 TO 50:FOR V=15 TO 0 STEP -3:
     SOUND 0,N(P),10,V:NEXT V:NEXT P
```

This is very powerful. You wouldn't think two TIN BRAINS like them would come up with something so nice. The more generalized version of this idea is:

```
 100 X=12:GOSUB 1000
1000 FOR V=15 TO 0 STEP -3:SOUND 0,N(X),10,V:NEXT V
1010 RETURN
```

Line 100 can be any line in your program. You can just define X as we did, or you can read it from a DATA statement, or from Disk or Tape. Then the subroutine plays the note by picking it from the array of 50 possibilities. By reading in notes and going to the subroutine, you can play tunes. All we need to do is develop a method to hold the notes for a time. This will give us Half notes, Whole notes, etc. This is done next in screens 12 to 17.

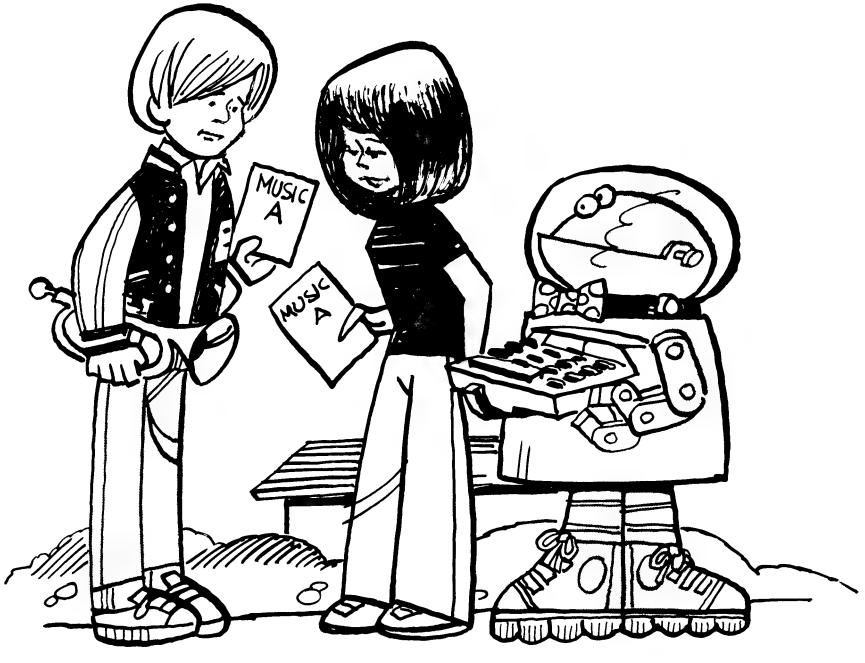
One last point in this lesson. We don't include the attack part of the loops that I went through all the trouble to explain before. The reason is that here a piano-type sound is desired, and by starting the DECAY loop at 15, we achieve INSTANT ATTACK (ie., very quick).

MAJRMINR (Major & Minor Chords)
Screens 18 to 25

This program expands on the idea of the 50 note array. Now you can play chords by using exactly the same numbers you did for notes (from the pitch chart .... remember)? Here are the BASIC program lines to do this. To see it in more detail, BREAK the program and look at lines 200 to 210:

```
 100 X=12:GOSUB 2000
2000 SOUND 0,N(X),10,14:SOUND 1,N(X-4),10,6:
     SOUND 2,N(X-7),10,6:SOUND 3,N(X-12),10,6:
     NEXT X:RETURN
```

If you have previously read in the note array (see actual examples in any of the song programs), this will play any chord that corresponds to the note you have chosen. In this case a 12 corresponds in the chart to C#. I'll tell you a secret. Orbie and Pixel invented this method as a homework assignment. You can bet they got an A+!



The KEYOFC, KEYATOC, and KEYDTOG Programs

These three programs illustrate chords by drawing notes on the screen, displaying the SOUND commands required to create each note, and executing these SOUND commands.

The KEYATOC (Keys of A, A#, B, C, and C#), and KEYDTOG (Keys of D, D#, E, F, F#, G, and G#), programs, use Graphics Mode 8. The KEYOFC program demonstrates the key of "C" only. It shows you a C Major chord, and the C Major scale. Try to understand how the KEYOFC program works by reading the program listing. It's not very different from the method we have been learning. Just the details have changed. The BASIC code is difficult to read because so many variables were used in order to save memory space. The variable X is used to store the horizontal screen coordinate while Y is used for vertical positioning. These programs are easy to understand when you run them. They're basically demonstrations to show how to combine graphics and music.

These programs do not teach music. They merely use the treble clef to show what many notes look like in sheet music form, while they play standard major chords.

Notes are drawn by specifying the X and Y coordinates, then using the subroutine beginning at line 160 to draw the note on the screen. Although seven PLOT and seven DRAWTO commands were used along with addition and subtraction, the notes are drawn rather quickly. Who says ATARI BASIC is slow?

The CHORDS12 program

Up until now, all of our music has been played using SOUND commands with a Distortion variable of 10. You may have noticed that there were no deep bass sounds. Using distortion 10, the lowest note possible really isn't very low.

So how do you get bass notes? I'm glad you asked. I just happened to include a program to demonstrate bass chords called CHORDS12. The key to deep bass sounds is that number 12 used in the distortion variable.

The program begins by playing a very low A Major Chord. Each time you press the START key, the next highest major chord is played, and the SOUND command used to create this note is displayed. Don't bother trying to memorize them, a separate chart of chords will be provided in a moment.

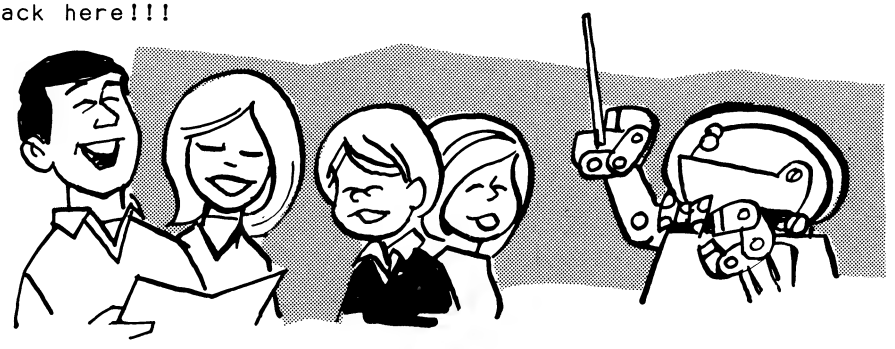
When the A Major chord is reached again, it will be one octave higher than when it started. When you press START this time, every pitch from 255 to 0 in distortion level 10 will be displayed on the screen, one at a time. This presents the notes in their undistorted state.

BASIC CHORDS

* CHORD *	DIST	FIRST	THIRD	FIFTH	DIST	FIRST	THIRD	FIFTH
A Major	10=	144	114	96	12=	75	60	51
A Minor	10=	144	121	96	12=	75	63	51
A# or Bb Major	10=	136	108	91	12=	72	57	48
A# or Bb Minor	10=	136	114	91	12=	72	60	48
B Major	10=	128	102	85	12=	67	52	45
B Minor	10=	128	108	85	12=	67	57	45
C Major	10=	121	96	81	12=	63	51	42
C Minor	10=	121	102	81	12=	63	52	42
C# or Db Major	10=	114	91	76	12=	60	48	40
C# or Db Minor	10=	114	96	76	12=	60	51	40
D Major	10=	108	85	72	12=	57	45	37
D Minor	10=	108	91	72	12=	57	48	37
D# or Eb Major	10=	102	81	68	12=	52	42	36
D# or Eb Minor	10=	102	85	68	12=	52	45	36
E Major	10=	96	76	64	12=	51	40	33
E Minor	10=	96	81	64	12=	51	42	33
F Major	10=	91	72	60	12=	48	37	31
F Minor	10=	91	76	60	12=	48	40	31
F# or Gb Major	10=	85	68	57	12=	45	36	30
F# or Gb Minor	10=	85	72	57	12=	45	37	30
G Major	10=	81	64	53	12=	42	33	28
G Minor	10=	81	68	53	12=	42	36	28
G# or Ab Major	10=	76	60	50	12=	40	31	26
G# or Ab Minor	10=	76	64	50	12=	40	33	26

DOE RAY ME

This program plays a song while displaying sing-a-long words on the screen. A 'walk' through this program will help you understand how to write your own BASIC songs so please refer to the listing at the end of this section. We are now going to go through the methods I have been teaching in detail, so if you are the type who just wants to play the songs, go ahead! Have fun while the rest of us are working hard to learn something. WE DON'T MIND. Hey, Everyone come back here!!!



First, a string called LINE$ and an array called 'N' -- which will store 50 pitches -- are DIMentioned. These pitches correspond to musical notes. LINE$ will store one line of words in our song. After the voices V0=0 (voice 0), V1=1 (voice 1), etc., are set, the program goes to line 100.

Why do we GOTO a line of DATA? Actually, we should have gone to line 120, but ATARI BASIC will bail us out and just fall through to line 120, which sends us off to a subroutine at line 21000. That routine just clears the screen, displays the heading, POKEs location 77 with a zero, and RETURNS. In case you don't know what that POKE does, it temporarily defeats ATARI's automatic color changing routine which is also known as attract mode.

RETURNing to line 120, we read data for 50 notes into the N (NOTE) array. The POKE 82,8 indents the left margin.

Before we start reading more data at line 210, it is important to understand the use of the "N" array and the subroutines found from line 30 through line 74.

Look at the DISTORTION LEVEL 10 PITCH CHART. The firs note is quite logically numbered 1. Its corresponding PIT is 14 and the musical note is C. This is a very high sour The higher the sound, the lower the NOTE# and PITCH val In our "N" array, N(1) contains a 14. The DATA in lines and 110 correspond to the PITCHes on the chart.

Why use NOTE numbers in an array when ATARI supplies PITCH values in their BASIC REFERENCE manual? I'm glad you asked. Start reading the PITCH values on the PITCH chart until you get to the number 21. What happened to 20? If you look further down the chart, you'll notice an increasing number of missing numbers. Now look down the column of NOTE #s. you will find 50 consecutive numbers. This provides us with a quick and easy way to let BASIC calculate chords when we supply only the base note of the desired chord.

The subroutine beginning at line 40 is the chord calculator; just supply it with the NOTE # in the variable "P". The routine assumes that P will be at least 8 and not greater than 50. P0 (PITCH 0) is then set equal to N(P). Then the program calculates the chord and stores the pitches in P1, P2, and P3. All 4 voices are turned on in line 42. Notice that VOICE 0 is set at a greater volume, and the three notes of our chord are played at a lower volume but equal to each other.

Line 50 is a WAIT routine. The value stored in the variable WAIT is POKEd into location 540. This location counts backwards to zero at the rate of 60 per second (JIFFIES). The program wastes time at line 52 until the countdown is completed. Then we turn off VOICE 0 only and RETURN.

So what does all this accomplish? In plain English, we played a melody note along with a chord, then turned off the melody note. The chord continues to play. The subroutine at line 60 is used to turn off ALL voices.

The subroutine at line 30 will change only the melody pitch, then go on to the WAIT routine.

The subroutine at line 70 turns on all four voices at equal volume, then decreases the volume gradually, until all sounds are off.

Now, where were we??? Ah yes, line 210 where we read LINE$, CHORD, P, WAIT:PRINT LINE$: and go off to the subroutine beginning at line 40. We are reading the DATA which begins at line 1000. We read the words, "DOE A DEER A FEMALE DEER" into a string then put it on the screen. We also read the number 49 into the variable CHORD, 37 into the variable P, and 45 into the variable WAIT.

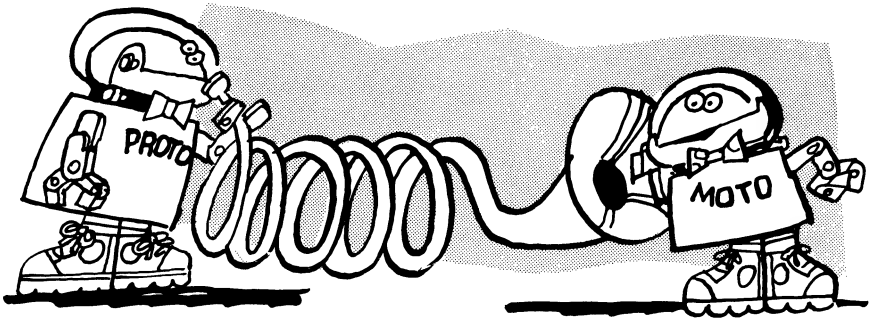
Remember, GOSUB 40 to play a melody note, calculate a chord, and kill some time. As I've said, the best way to learn from examining someone else's program is by acting as if you were the computer. Follow the instructions, and see what you, or the computer, will do. Let's try it.

I'll be the computer this time. I have just read the data as I was instructed to do in line 210, and now I'm at the subroutine at line 40. I am told to make P0=N(P). I just read DATA and set the value of P=37. I look up the value of N(37) and see that it is 121. I set P0=121.

My next instruction is to make P1=N(CHORD). I read the value of CHORD in line 210 and know that CHORD=49. I look up the value of N(49) which is 243, and set P1=243. P2 must be set to the value stored in N(45) and P3 must be set to the value stored in N(42). P2=193 and P3=162.

I turn on all four voices as indicated in line 42 then POKE the number 45 into my memory location 540. At line 52 I look at the value stored in memory location 540 and compare it to zero. It's not 0 so I check it again. Each time I check that location, its value is less than it was last time I looked, but it's not zero so I keep checking. I'm getting bored.

Finally, I find a zero and go on to line 54. I turn off the sound of Voice 0 then RETURN from this subroutine to line 220.



Now it's your turn. Continue through the program doing what I just did. If you start getting confused, take a pencil and write down values as you read and change them accordingly. You should soon understand what I've put your computer through to play this simple song.

The logic in this program is not suitable for all songs. You will have to make minor modifications for different tempos, or if other than standard MAJOR chords are required. This program demonstrates one way to play a simple song and an easy method of finding the notes of a chord. Don't think you can just add a few lines of DATA and create the Nutcracker Suite.

If you just want to enter music, then see it in sheet music form while it is played, I'd recommend ATARI's MUSIC COMPOSER. If you'd like to see your melody as it would be played on a piano, or play your keyboard as if it were a piano, the PLAYER PIANO section is what you want.

```
0 REM DORAYME (c) 1981 by Jerry White 11/1/81
10 DIM LINE$(40),N(50):V0=0:V1=1:V2=2:V3=3:GOTO 100
30 SOUND V0,N(P),10,14:GOTO 50
40 P0=N(P):P1=N(CHORD):P2=N(CHORD-4):P3=N(CHORD-7)
42 SOUND V0,P0,10,14:SOUND V1,P1,10,6:SOUND V2,P2,10,6
45 SOUND V3,P3,10,6
50 POKE 540,WAIT
52 IF PEEK(540)<>0 THEN 52
54 SOUND V0,0,0,0:RETURN
60 FOR OFF=0 TO 3:SOUND OFF,0,0,0:NEXT OFF:RETURN
70 P0=N(P):P1=N(CHORD):P2=N(CHORD-4):P3=N(CHORD-7)
72 FOR DECAY=8 TO 0 STEP -1:SOUND V0,P0,10,DECAY
73 SOUND V1,P1,10,DECAY:SOUND V2,P2,10,DECAY
75 SOUND V3,P3,10,DECAY
80 NEXT DECAY:RETURN
100 DATA 14,15,16,17,18,19,21,22,23,24,26,27,29,31,33,35,37
104 DATA 40,42,45,47,50,53,57,60,64,68,72,76,81,85,91,96
110 DATA 102,108,114,121,128,136,144,153,162,173,182,193
115 DATA 204,217,230,243,255
120 GOSUB 21000:FOR X=1 TO 50:READ IT:N(X)=IT:NEXT X
200 TRAP 19000:POKE 82,8:? " ":?
210 READ LINE$,CHORD,P,WAIT:? LINE$:GOSUB 40
220 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60
225 WAIT=10:GOSUB 50
230 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
240 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60
245 WAIT=30:GOSUB 50
250 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
260 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60
265 WAIT=10:GOSUB 50
270 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
280 FOR ME=1 TO 6:READ P,WAIT:GOSUB 30:NEXT ME:GOSUB 60
285 WAIT=30:GOSUB 50
290 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
300 FOR ME=1 TO 5:READ P,WAIT:GOSUB 30:NEXT ME
310 READ CHORD,P,WAIT:GOSUB 40:GOSUB 60:WAIT=30:GOSUB 50
320 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
330 FOR ME=1 TO 5:READ P,WAIT:GOSUB 30:NEXT ME
340 READ CHORD,P,WAIT:GOSUB 40:GOSUB 60:WAIT=30:GOSUB 50
350 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
360 FOR ME=1 TO 5:READ P,WAIT:GOSUB 30:NEXT ME
370 READ CHORD,P,WAIT:GOSUB 40:GOSUB 60:WAIT=10:GOSUB 50
380 READ LINE$,CHORD,P,WAIT:? :? LINE$:GOSUB 40
390 READ P,WAIT:GOSUB 30
400 READ CHORD,P,WAIT:GOSUB 40
410 READ P,WAIT:GOSUB 30
420 READ CHORD,P,WAIT:GOSUB 40
430 READ P,WAIT:GOSUB 30
440 READ CHORD,P,WAIT:GOSUB 40
450 GOSUB 60:WAIT=10:GOSUB 50:FOR DECAY=15 TO 0 STEP -0.5
455 SOUND V0,N(1),10,DECAY:NEXT DECAY
460 GRAPHICS 18:? #6:? #6;"    major chords"
510 FOR ME=1 TO 8:READ CHORD,P,LINE$:POSITION ME*2,10-ME
520 ? #6;LINE$:GOSUB 70:NEXT ME
```

```
 530 FOR ME=8 TO 1 STEP -1:READ CHORD,P,LINE$
 53ɔ POSITION ME*2,10-ME:? #6;LINE$:GOSUB 70:NEXT ME
 540 WAIT=15:GOSUB 50:POSITION 4,11:? #6;"PRESS"
 550 FOR DECAY=15 TO 0 STEP -0.5:SOUND V0,N(6),10,DECAY
 555 NEXT DECAY
 56ʊ WAIT=15:GOSUB 50:POSITION 10,11:? #6;"START"
 570 FOR DECAY=15 TO 0 STEP -0.5:SOUND V0,N(1),10,DECAY
 58ʊ NEXT DECAY
 600 SETCOLOR 0,PEEK(20),10:IF PEEK(53279)<>6 THEN 600
 700 GRAPHICS 18:SETCOLOR 0,1,10:SETCOLOR 1,11,12
 705 SETCOLOR 3,4,12
 710 ? #6:? #6:? #6;"    PRESS option":? #6;"      TO RERUN"
 720 WAIT=60:GOSUB 50
 730 ? #6:? #6;"    PRESS start":? #6;"    TO CONTINUE"
 740 IF PEEK(53279)=3 THEN RUN
 750 IF PEEK(53279)=6 THEN 900
 760 GOTO 740
 900 RUN "D:KEYOFC"
1000 DATA DOE A DEER A FEMALE DEER
1010 DATA 49,37,45,35,15,33,45,37,15,33,30,37,30,33,45
1020 DATA RAY A DROP OF GOLDEN SUN
1030 DATA 42,35,45,33,15,32,15,32,15,33,15,35,15,32,90
1040 DATA ME A NAME I CALL MYSELF
1050 DATA 49,33,45,32,15,30,45,33,15,30,30,33,30,30,45
1060 DATA FA' A LONG LONG WAY TO RUN
1070 DATA 44,32,45,30,15,28,15,28,15,30,15,32,15,28,90
1080 DATA SEW A NEEDLE PULLING THREAD
109ʊ DATA 49,30,45,37,15,35,15,33,15,32,15,30,15,44,28,90
1100 DATA LA A NOTE TO FOLLOW SEW
1110 DATA 44,28,45,35,15,33,15,31,15,30,15,28,15,42,26,90
1120 DATA TEA A DRINK WITH JAM AND BREAD
1130 DATA 42,26,45,33,15,31,15,29,15,28,15,26,15,49,25,90
1140 DATA THAT WILL BRING US BACK TO DOE
1150 DATA 49,25,15,26,15,44,28,30,32,30
1170 DATA 42,26,30,30,30,49,25,90
1200 DATA 49,37,C,47,35,D
1220 DATA 45,33,E,44,32,F
1240 DATA 42,30,G,40,28,A
126ʊ DATA 38,26,B,37,25,C
1300 DATA 37,25,c,38,26,b
1320 DATA 40,28,a,42,30,g
1340 DATA 44,32,f,45,33,e
1360 DATA 47,35,d,49,37,c
21000 GRAPHICS 0:SETCOLOR 2,9,0:SETCOLOR 4,9,0
21005 SETCOLOR 1,9,12:POKE 752,1:POKE 82,2:POKE 83,39
21007 POKE 201,7
21010 ? " ":? ,"┌──────────────────────┐"
21020 ? ," IMAJOR CHORD HARMONYI  "
21030 ? ," └──────────────────────┘ "
21040 POKE 77,0:RETURN
```

The SING-A-LONG programs

This group of three sing-a-long programs consists of BIRTHDAY, JINGLE, and SILENT. Each plays a song while the words to the song are displayed on the screen. BIRTHDAY plays the Happy Birthday Song, JINGLE plays Jingle Bells, and SILENT plays Silent Night. To understand how they work, please refer to the listing for BIRTHDAY at the end of this section.

BIRTHDAY uses four string variables. NAME$ holds the name of the person to whom our song will be dedicated. WORD$ holds the words of the song as they are read from DATA statements. DISP$ holds the words as they will be displayed on the screen. BLANK$ holds a string of twenty blanks and is used to center the words on the screen. Each string is DIMensioned at twenty characters, the maximum number that can displayed on the screen at any given time.

The words are displayed in GRAPHICS MODE 18 which is mode 2 without a text window. A single line of characters cannot be greater than twenty.

The program begins in GRAPHICS MODE 0 and you must enter the name of the birthday person. String DIMensions are made in lines 20 and 40, then we skip over some subroutines and begin reading data at line 300.

For each change of sound and display, read a pitch for each of the four voices and store this data in the variables V0, V1, V2, and V3. This program also reads data to be stored in the variables HOLD, SWITCH, and WORD$. HOLD tells the program how long to hold the current sounds and display in 60ths of a second. SWITCH is used to note special functions such as when to display the birthday person's name instead of the words found in our DATA. WORD$ stores the words as they are read from the DATA. It's most important to note that each line of data MUST contain seven items, separated by commas.

After each set of seven items of data is read, the SWITCH variable is checked to see if something special should be done. In this program, I check for a 1, 3, or a 9. If SWITCH=3 then I want to display the person's name and not the word, "NAME". If SWITCH=1 then I want to clear the screen, and go on to line 360. If SWITCH=9 then the song is over and I want to GOTO line 9000.

The routine in lines 360 and 370 does some string manipulation and eventually winds up with the words as they will be displayed in the string DISP$. This string contains the leading and trailing blanks necessary to display our words neatly centered on the screen.

In line 380, the words are put on the screen, GOSUB 100, then it goes back to line 300 to get more DATA and do it all over again.

The subroutine at line 100 POKEs the value stored in the variable HOLD, into the countdown timer at location 540. Then all four voices are turned on using the pitch values just read. Notice that voice zero (SOUND 0) is twice as loud as any other. This is the melody voice. It is louder so that it stands out. You may change volumes as you see fit. Just remember that you shouldn't exceed a total volume of 32 or distortion and decrease in volume will occur.

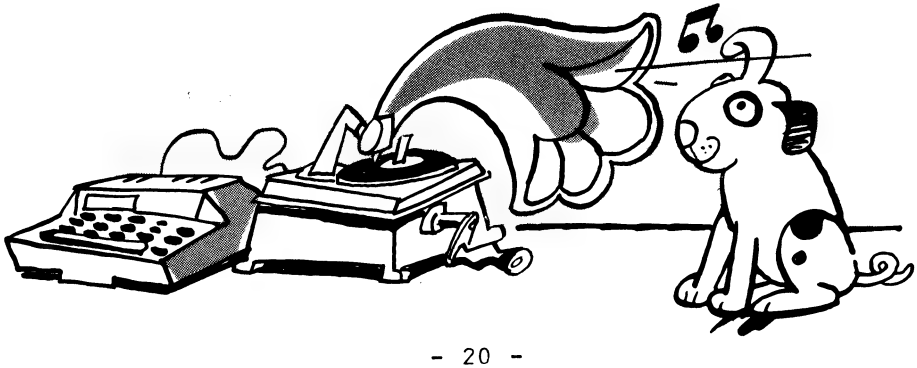
Line 101 loops until the countdown timer has gone back to zero. The four voices turn off in line 102 and RETURN from this subroutine.

The changes in the color of the words on the screen is coordinated with the current syllable being sung. This is done by keeping the syllables yet to be sung in inverse video. A quick look at the words to our BIRTHDAY song will show you how this works.

The hard part of entering the DATA is determining the pitches for the four voices. You either have to use the old trial and error method, or read sheet music and enter the data accordingly. There is, however, no need to use four part harmony. To use only one voice, remove the SOUND commands for voices 1, 2, and 3 in lines 100 and 102, and the reading of V1, V2, and V3 from line 300.

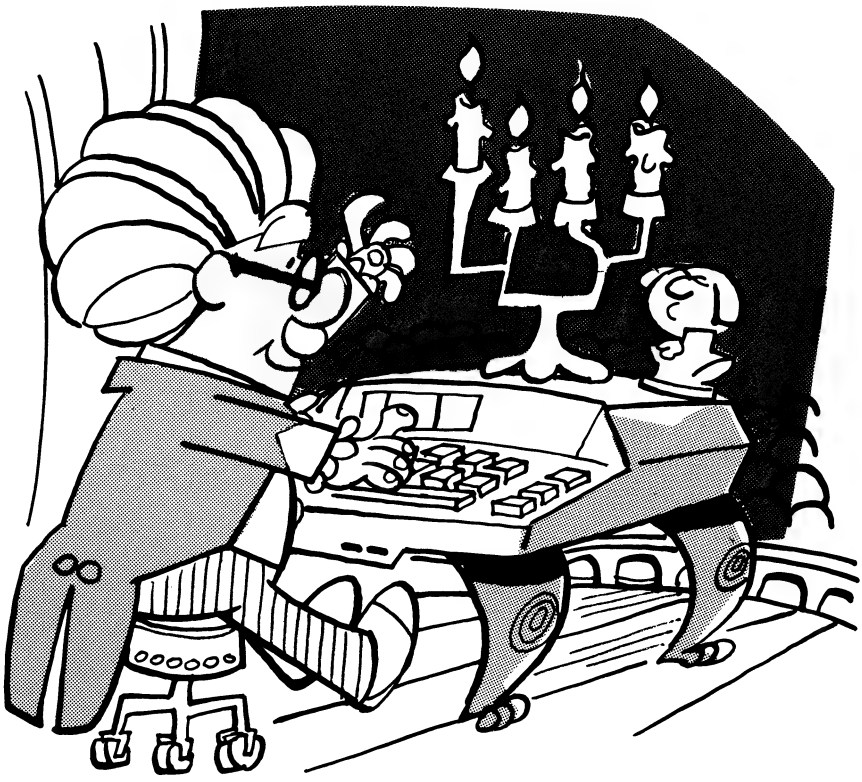
You may notice that in most cases only two different pitches are used in the birthday song. Only the final two notes actually use four part harmony. Throughout the rest of the song, I'm putting the same melody note into V0 and V1, and the same note, one octave lower, into V2 and V3.

When a REST (a period with no sound at all) is needed , we can specify pitch zero. This is almost cheating but it works because that pitch is beyond human hearing. If you have a dog, Rover may begin to sing during rests!



Trial and error will be the best teacher on the HOLD variable. There is a method to my madness. The reason I use numbers like "line 19 and 39" instead of 20 and 40 was to allow that extra jiffy to shut off sounds and get more data, etc. As it turns out, I probably should have allowed a bit more time since I'm also checking switches, and centering text on the screen. The method I used does seem to work, although I'm sure many of you will eventually find better ways.

Here are a few tips from my own trial and error method of entering data to the three songs in this package. Don't enter more than one or two lines of DATA without testing. You test by running the program and listening. Use this song as a guide. It is very slow. The whole notes last for 80 jiffies or one and one-third seconds. To get the values of halfnotes and quarternotes, simple division does the trick. Half of 80 is 40, a quarter of 80 is 20. Subtracting 1 provides the durations 79, 39, and 19. If you come out to a noninteger, make like ATARI BASIC and round off to the lower number.



BIRTHDAY

```
0 REM HAPPY BIRTHDAY (c) 1982 BY JERRY WHITE [6/3/82]
20 GRAPHICS 0:DIM NAME$(20):? :? "BIRTHDAY SONG PROGRAM:":?
30 ? "ENTER NAME:";:INPUT NAME$
40 DIM WORD$(20),DISP$(20),BLANK$(20)
50 BLANKS$="                    ":GOTO 300
60 POKE 540,HOLD:SOUND 0,V0,10,8:SOUND 1,V1,10,4
70 SOUND 2,V2,1,4:SOUND 3,V3,10,4
101 IF PEEK(540)<>0 THEN 101
102 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 2,0,0,0:SOUND 3,0,0,0
105 RETURN
300 READ V0,V1,V2,V3,HOLD,SWITCH,WORD$
310 IF SWITCH=3 THEN WORD$=NAME$:GOTO 360
330 IF SWITCH=1 THEN GRAPHICS 18:GOTO 360
340 IF SWITCH=9 THEN 9000
360 LW=LEN(WORD$):DISP$=BLANK$
365 IF LW>18 THEN DISP$=WORD$:GOTO 380
370 FP=11-INT(LW/2):DISP$(FP,FP+LW-1)=WORD$
380 POSITION 0,4:? #6;DISP$:GOSUB 100:GOTO 300
500 DATA 121,121,243,243,19,1,HAPPY BIRTHDAY
510 DATA 121,121,243,243,19,0,HAPPY BIRTHDAY
520 DATA 108,108,217,217,39,0,HAPPY BIRTHDAY
530 DATA 121,121,243,243,39,0,HAPPY BIRTHDAY
540 DATA 91,91,182,182,39,1,TO YOU
550 DATA 96,96,193,193,79,0,TO YOU
600 DATA 121,121,243,243,19,1,HAPPY BIRTHDAY
610 DATA 121,121,243,243,19,0,HAPPY BIRTHDAY
620 DATA 108,108,217,217,39,0,HAPPY BIRTHDAY
630 DATA 121,121,243,243,39,0,HAPPY BIRTHDAY
640 DATA 81,81,162,162,39,1,TO YOU
650 DATA 91,91,182,182,79,0,TO YOU
700 DATA 121,121,243,243,19,1,HAPPY BIRTHDAY
710 DATA 121,121,243,243,19,0,HAPPY BIRTHDAY
720 DATA 60,60,121,121,39,0,HAPPY BIRTHDAY
730 DATA 72,72,144,144,39,0,HAPPY BIRTHDAY
740 DATA 91,91,182,182,39,1,DEAR
750 DATA 96,96,193,193,39,3,NAME
760 DATA 108,108,217,217,79,3,NAME
770 DATA 0,0,0,0,19,3,NAME
800 DATA 68,68,136,136,19,1,HAPPY BIRTHDAY
810 DATA 68,68,136,136,19,0,HAPPY BIRTHDAY
820 DATA 72,72,144,144,39,0,HAPPY BIRTHDAY
830 DATA 91,91,182,182,39,0,HAPPY BIRTHDAY
840 DATA 81,96,121,162,39,1,TO YOU
850 DATA 91,121,144,182,79,0,TO YOU
8000 DATA 0,0,0,0,0,9,END
9000 GRAPHICS 18:SETCOLOR 2,4,8:SETCOLOR 0,12,6
9050 SETCOLOR 4,4,0
9100 POSITION 3,4:? #6;"HAPPY BIRTHDAY"
9200 FOR HOLD=1 TO 500:NEXT HOLD:GOTO 30000
30000 GRAPHICS 0:POKE 752,1:POKE 710,48:POKE 82,2:POKE 201,9
30006 RUN "D:SILENT"
```

THE SONG WRITER PROGRAM

This program offers you the opportunity to compose your own songs on the ATARI computer. SONGRITE provides a skeleton program to which you must add your own DATA statements. A sample DATA statement has been included. Remember, the READ statement in line 1280 requires exactly seven parameters to produce each note so the DATA statements you provide need to account for all of them.

The first four parameters are the actual pitches for each of the four voices. You can consult the pitch chart in this manual to get specific notes or you can just experiment with any number between 0 and 255.

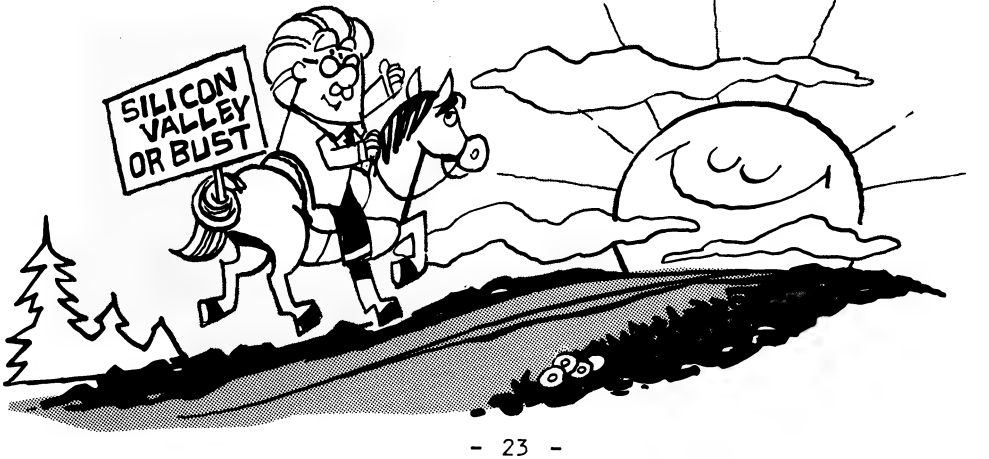
The fifth parameter is HOLD, the variable showing how long to hold the note. The number 60 would hold the note for one second; the number 30 a half second,etc.

The sixth parameter is SWITCH. In this program, the switch is tested right after the READ statement to see if SWITCH is equal to 9. If it is the program ENDs. An alternative is to test for 9 and IF SWITCH=9 THEN GRAPHICS 0:LIST. This technique is very useful for writing and repeatedly testing the program and getting back to the listing after each test. Any number of switches can be include in your program.

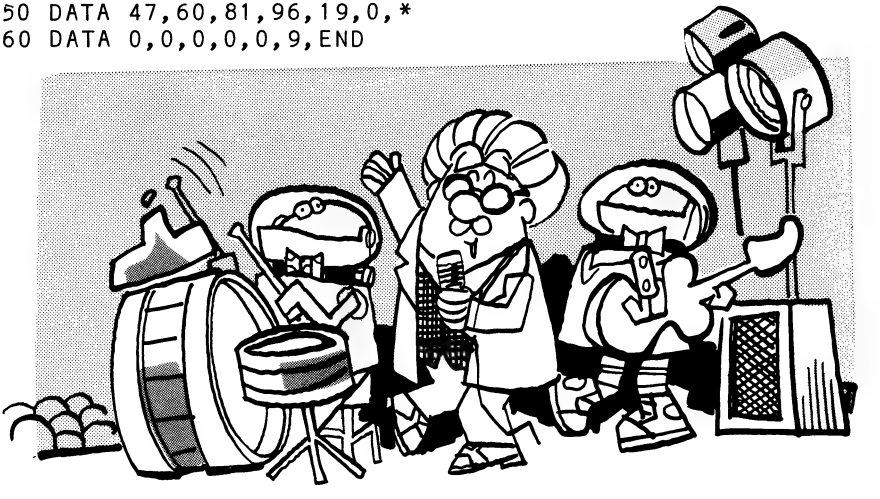
The seventh parameter is WORD$, the string variable for the actual words to appear on the screen as the music is played. Any word or words up to 40 characters long will fit.

It's helpful to study the listings for SILENT, JINGLE, and BIRTHDAY to see how these songs were written. These programs demonstrate the use of switches as well as different graphics techniques which you may want to include in your own programs. Above all, have fun.

HAPPY TUNES TO YOU .....



```
1000 REM ****************************
1010 REM *    SONG WRITER  (6/3/82) *
1020 REM *  (C) 1982 by Jerry White *
1030 REM ****************************
1040 GRAPHICS 0:POKE 752,1
1050 LIST 1000,1030:?
1060 ? "   ENTER DATA BEGINNING AT LINE 500":?
1070 ? "         DELETE LINES 1050 THRU"
1075 ? "           1130 BEFORE TESTING ":?
1080 ? "           TO RUN PLAYER PIANO"
1090 ? "               PRESS START   ":?
1100 ? "         TO ADD DATA PRESS SELECT"
1110 IF PEEK(53279)=6 THEN RUN "D:PIANODSK"
1120 IF PEEK(53279)=5 THEN POKE 752,0:END
1130 GOTO 1110
1140 DIM WORD$(40),DISP$(40),BLANK$(40)
1150 BLANK$="
1160 GOTO 1280
1170 REM PLAY THE NOTE
1180 POKE 540,HOLD
1190 SOUND 0,V0,10,8
1200 SOUND 1,V1,10,4
1210 SOUND 2,V2,10,4
1220 SOUND 3,V3,10,4
1230 IF PEEK(540)<>0 THEN 1230
1240 SOUND 0,0,0,0:SOUND 1,0,0,0
1250 SOUND 2,0,0,0:SOUND 3,0,0,0
1260 RETURN
1270 REM READ THE DATA
1280 READ V0,V1,V2,V3,HOLD,SWITCH,WORD$
1290 IF SWITCH=9 THEN POKE 752,0:END
1300 LW=LEN(WORD$):DISP$=BLANK$
1310 IF LW>38 THEN DISP$=WORD$:GOTO 1330
1320 FP=21-INT(LW/2):DISP$(FP,FP+LW-1)=WORD$
1330 POSITION 0,10:? DISP$:GOSUB 1180:GOTO 1280
1340 DATA 47,60,81,96,19,1,JINGLE BELLS
1350 DATA 47,60,81,96,19,0,*
1360 DATA 0,0,0,0,0,9,END
```



PLAYER PIANO

Player Piano provides an exciting opportunity for those who have additional memory (24K for cassette users - 32K for diskette users) to practice what you've learned by turning your keyboard into a mini-piano. A menu option allows you to create your own songs, SAVE or LOAD song data files using cassette or diskette, fix or change up to 400 notes in memory, and play all or part of a song. The screen displays the keyboard and indicates each key as it is played from a data file or the notes you type. You don't have to be a musician to enjoy this educational and entertaining program, just have an ATARI.

P.S. Yes, it is the same program APX sells for $22.95 by itself! Aren't we nice?!

PLAYER PIANO

Special Function Keys

Since this program uses the keyboard exclusively, special care has been taken to guard against user errors. All special function keys except ESC (escape) and DELETE BACK S are not used, and all but SYSTEM RESET are software disabled, so if you accidently hit the wrong key nothing will happen.

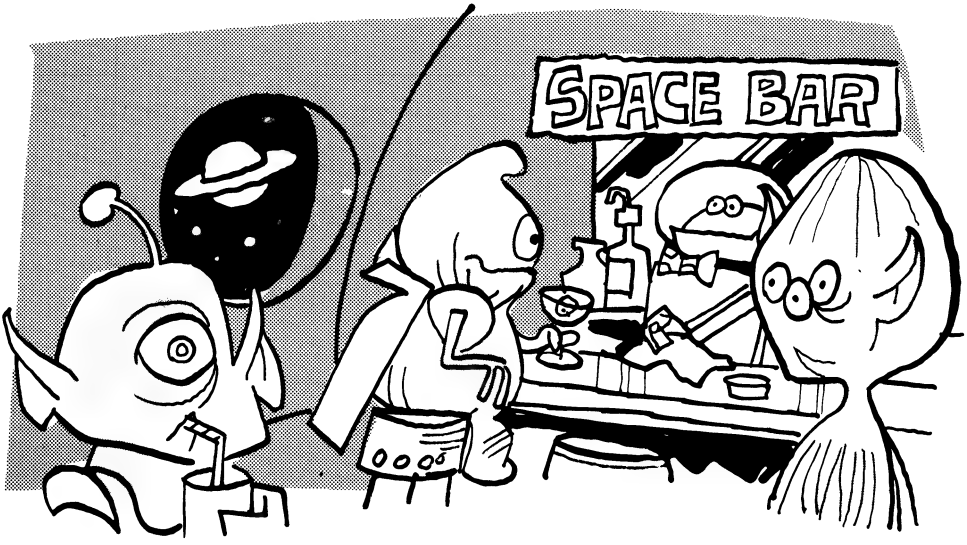
The ESC key can be used at any time to escape from the current option or menu selection. In other words, if you don't see what you want, press ESC key.

The DELETE BACK S key is used to backspace and erase the previous key displayed in the text window at the bottom of the screen display. This key works without the simultaneous holding of the CTRL key. The CTRL key is not used by this program and all typing is to be done in normal video upper case.

If you mistakenly press the Atari reverse video or lower case keys, the program will automatically reset to normal upper case when the next key is pressed.

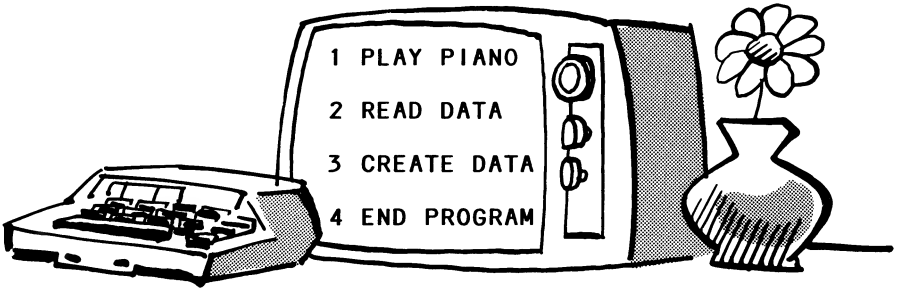
PIANO keyboard keys are displayed in the text window at appropriate times. The center two rows activate musical notes. All keys in the third row from the top, from A through * are used to play the natural--white key--notes of the PIANO keyboard. Keys on the second row from the top are used as indicated in the text window to play the sharp and flat--black key--notes of the PIANO keyboard.

The SPACE BAR is used to create a REST or shut off the sound. For example, while in PLAY PIANO mode, typing the A key will play the lowest--or leftmost--note on the piano display. You can get a very short, quick note by typing A then pressing the SPACE BAR immediately.



USING THE PROGRAM

After a brief introductory display, the screen will display the following options:



Begin by typing the number 1. Throughout this program, you will not have to press the RETURN key when a single key response is required. In this case, you had to type a single digit number. The program understands this and continues on about its tasks immediately. If a response could be more than one keystroke, the program will wait until you press the RETURN or ESC keys before it continues. If you do press the RETURN key when you didn't really have to, no harm is done.

Once the PIANO is drawn on the screen, the text window at the bottom will display the keys on your keyboard that correspond to the notes on the PIANO. After a little gold musical note jumps across the keys playing them for you, it's your turn. Experiment for a while to get used to the keyboard and remember to include the SPACE BAR. The SPACE BAR is most important since it shuts off the sound and permits the playing of very short notes.

Let me call your attention to the chart labeled PLAYER PIANO FILEDUMP and titled BIRTHDAY.The BASIC utility program FILEDUMP was used to create this chart.More on that later. For now, we will just use the chart to enter the BIRTHDAY song.

To do this, we must use the ADD DATA TO MEMORY or CREATE A NEW DATA FILE options. In this case it makes no difference which one we use since there is no data in memory at this time. If we already had data in memory, the ADD DATA TO MEMORY option would add your new notes to the end of the data file in memory. The CREATE A NEW DATA FILE option erases the data file in memory so we can start again from note number 1.

This program contains many options. In order to create a data file, we must ESCape from the PLAY PIANO mode so press the ESC key. You now have four options in the text window:

TYPE 1 TO ADD DATA TO MEMORY
TYPE 2 TO SAVE MEMORY DATA
TYPE 3 TO PLAY MEMORY DATA
TYPE 4 TO FIX MEMORY DATA

We could choose option number 1 right now but press the ESC key again to see another list of four options.

TYPE 1 TO PLAY KEYBOARD PIANO
TYPE 2 TO READ A DATA FILE
TYPE 3 TO CREATE A NEW DATA FILE
TYPE 4 TO END THIS PROGRAM

Type the number 3 to CREATE A NEW DATA FILE.

The text window will ask you to TYPE DURATION (1-120) RETURN. Look at NOTE #1 on the FILEDUMP chart. You will see four columns. Do not be concerned with the second column at all for now. We are interested in finding the duration of NOTE #1 which is 30. The duration is a number showing how many 1/60'ths of a second the note is held. By typing a duration of 30, we are saying, "Hold this note for 30/60 or half a second." Type 30 then press RETURN. Note that we had to press the RETURN key because the program can't tell if you will be typing more than one key or not.

Now we are asked to type NOTE #1. We type this note as if we were back in the PLAY PIANO mode. The rightmost column on the chart shows the key we need which is D. If all went well, that's one note down and 27 to go.

If you make a mistake, don't worry about it. There is an option for fixing data. You can always use the ESC key and enter the FIX DATA IN MEMORY option. That mode will ask you which note number you'd like to fix. Then type the appropriate note number and press RETURN. You can reenter the note or, once again, use the ESC key.

Here's a helpful hint. Before you enter too many "notes", it's very helpful to replay your notes before more data is added. If you do make a mistake, it's much easier to find and correct while it's fresh in your human memory. Enter the first six notes then press the ESC key.

Let's PLAY what we have so far. If you have the right list of options on the screen, number 3 should be PLAY MEMORY DATA. If not, press the ESC key again. Now type the number 3. The display should tell you that we have 6 notes in memory and ask us which one we would like to start with. Now replay the entire 6 notes by typing the starting number 1 and the ending number of 6 and press RETURN. If they sound like the first six notes of Happy Birthday to you, continue adding data. If not, find out which one is wrong and use the fix option to correct it. You can stop a song while it is being played by pressing any key.

On the FILEDUMP chart, NOTE #7 is SP. SP means SPACE BAR. Between the sixth and eighth notes is a full second rest, so type a duration of 60 and press the space bar for NOTE #7.

You may SAVE MEMORY DATA on cassette or diskette at any time by selecting that option number. Type C for cassette or D for diskette. When using cassette, be sure to use a blank tape and note the position number on your program recorder. Be sure to press PLAY and REC buttons before you press RETURN on the keyboard to begin saving your data.

If you are using a diskette, you must specify the file name. Be sure to use a diskette with the write enable notch uncovered, and make sure there are enough free sectors to store your data file. Disk drive #1 is assumed so you need only specify the filename. It's also a good idea to use a filename extention such as .DAT for Data file or .PPM for Player Piano Music file. The Birthday song could therefore be saved on diskette by typing D when asked if you are using Cassette of Diskette, then typing the filename BIRTHDAY and pressing RETURN. Any valid file name will do but remember that if you already have that filename on the diskette, the old file will be deleted before the new file is written. Do NOT write onto the original MASTER diskette.

You can also read song files such as BALLGAME, from cassette or diskette. Note that reading a file will replace any data currently in memory. If you want to keep the file in memory, save it before reading in another file.

This program has been designed to store up to 400 notes in memory.

SELECTING DURATION

Look at the FILEDUMP chart of the BIRTHDAY song once again. Notice the duration of each note is either 30 or 60, meaning notes having a duration of 60 will be held for one second and notes having a duration of 30 will be held for one half second.

This song is used as a demonstration because it is universally known and very simple. It is also very slow. A note held for a full second is rare!

Most songs you enter will probably be at a faster tempo. Tempo is based on what is called a whole note. In the BIRTHDAY song, the whole notes are those with the duration of 60. The others would all be half notes, quarter notes, and eighth notes. If the BIRTHDAY song required quarter notes, the duration would be 15, or one quarter of 60.

O.K., how do we divide 15 in half for the eighth note? It's really quite simple. Just drop any fractions. So in this case the duration would be 7.

You will find that there is a limit of around 6 for the shortest possible note this program can replay. It takes about a tenth of a second for it to interpret the data into a sound, position the little gold note on the proper key, and display the note number in the text window. As an example, try this: specify a duration of 6 for one note, then specify a duration of 1 for the following note, both will be the same duration.

Before you begin to enter a song of your own, think about its tempo. If it's slow like the BIRTHDAY song, use 60 as your whole note and calculate the shorter notes based on fractions of the whole note. If the tempo is faster, try 40 as the whole note duration, 20 for the half note, etc.

FILE LAYOUT AND THE FILEDUMP PROGRAM

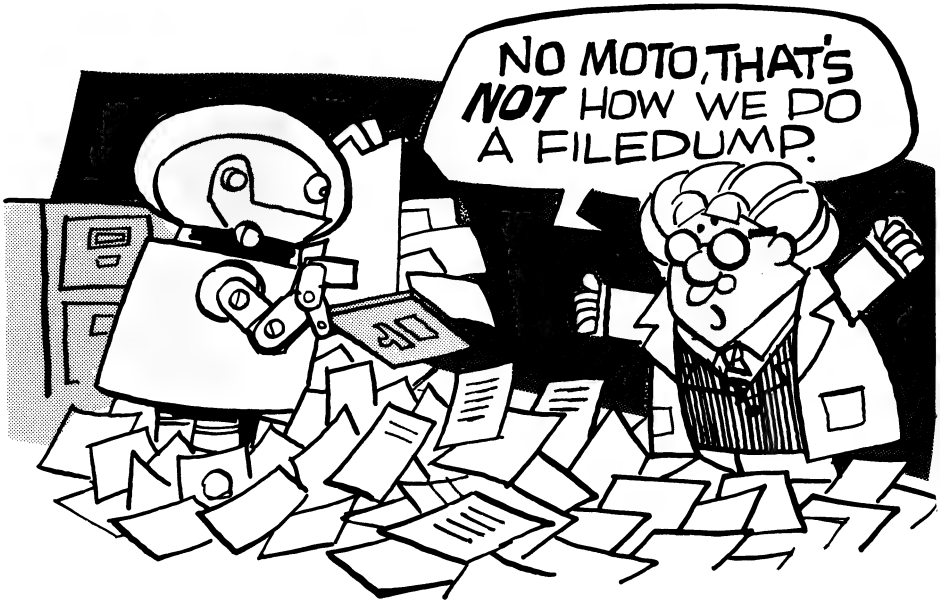
The FILEDUMP program will come in handy when you write longer and more complex songs. Enter the program exactly as it appears in the listing. Make sure that there is no program in memory before you begin. One sure way to remove any existing program from memory is by giving the BASIC command NEW.

You don't need a printer. FILEDUMP has been designed to let you decide to print or display the files on the screen. It is also set up so you can input cassette or diskette song files.

PLAYER PIANO song files are simply a data file of numbers. There are two numbers for each note in the file. The first is called the KEYCODE and the second is the duration.

The first two numbers in the file are not for note number one. The very first number specifies the total number of notes in the file. The second number of this first set is a dummy zero. After the first set of two numbers is read, PIANO and FILEDUMP know exactly how many more sets of numbers to read.

With one exception, the keycode is the ATASCII value of the key pressed minus 42. Do not confuse this with the computer's internal keycodes. The keycodes used by PIANO and FILEDUMP are generated specifically for these programs.



There always seems to be an exception to every rule. The SPACE BAR keycode is 2. Don't ask why! WHY? I was afraid you'd ask that! This was done to save a little memory in the PIANO program. It keeps the array of keycodes and their corresponding sounds and screen positions smaller.

If you do some programming on your own, you may wish to write a little program for yourself to increase or decrease the duration of each note by a given percentage. This would permit the changing of tempo in a song. Use the FILEDUMP program as a model.

There are twenty notes on this PLAYER PIANO. As previously mentioned, the SPACE BAR has the keycode of 2. In case you are interested, here are the keycodes for all the notes from lowest (far left A), to highest (far right E).

A	A#	B	C	C#	D	D#	E	F	F#	G	G#	A	A#	B	C	C#	D	D#	E
23	45	41	26	40	28	42	29	30	43	32	31	33	37	34	17	3	1	19	0

PLAYER PIANO FILEDUMP

BIRTHDAY

NOTE#	KEYCODE	DURATION	KEY
1	26	30	D
2	26	30	D
3	28	60	F
4	26	60	D
5	26	60	H
6	29	60	G
7	2	60	SP
8	26	30	D
9	26	30	D
10	28	60	F
11	26	60	D
12	32	60	J
13	30	60	H
14	2	60	SP
15	26	30	D
16	26	30	D
17	17	60	I
18	33	60	K
19	30	60	H
20	29	60	G
21	28	60	F
22	2	60	SP
23	37	30	0
24	37	30	0
25	33	60	K
26	30	60	H
27	32	60	J
28	30	60	H

ALSO FROM EDUCATIONAL SOFTWARE

PROTOTYPE'S ADVENTURES
Ages 4 to 8

by Stan Gilbert. A delightful collection of 8 NON-VIOLENT and NON-COMPETITIVE games written especially for young children. Professor Von Chip's robot, Proto Type, will guide your child through activities such as catching free MARSHMALLOWS from friendly aliens, electronic COLORING, JUMP ROPE, and a fun game called ALPHA BLOCKS. Proto even shows them where he lives and works. The storybook, shaped like Proto Type, has cartoons for your child to color that don't need a computer to be fun. 16K-Tape/24K-Disk -- $19.95

DOG DAZE

by Gray Chang. A fast action arcade style game where the players are puppies fighting for control of the local fire hydrants. Each pooch races to claim the hydrants either by touching it first or by shooting his bone at it. And doglike, if a dog comes too close to another dog's hydrant, he's compelled to stop and sniff, losing valuable time. Each dog must be careful of cars or the game could end rather suddenly! Great fun for all ages! Written in efficient machine language. 8K-Tape/24K-Disk -- $16.95

SPACE GAMES

by Randy Massey. Three exciting games for one very affordable price. With all the fun you'll have playing ALIEN, SURVIVE, and ROBOT ATTACK you can't lose even if you get turned into space dust! 16K-Tape/32K-Disk -- $24.95 for all three.

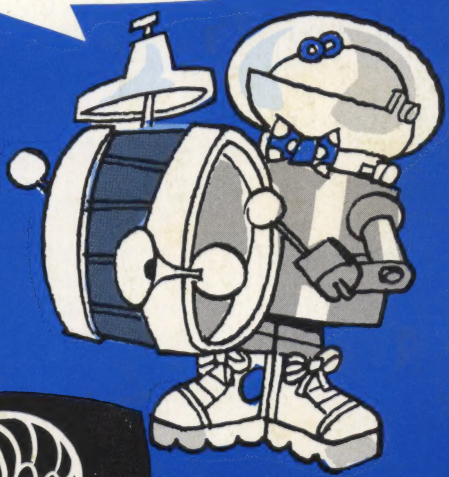
IN THE UNLIKELY EVENT
YOU RECEIVE A DEFECTIVE PRODUCT
RETURN IT TO US FOR A
PROMPT REPLACEMENT OR REFUND

SOUND & MUSIC

TRICKY TUTORIAL #6 (TM)

This program starts with the simple sound statement, but progresses to chords and complete songs. All of the material can be used by a beginner, yet if it is studied, you will learn many of the tricks that Jerry White puts into his other musical programs for the ATARI (Name That Song, Player Piano, My First Alphabet's tunes). **MUSIC FROM BASIC BECOMES ALMOST EASY!** Player Piano is included FREE with this package (24K/32K required)!

This program requires 16K for tape users or 24K for disk, and a Basic cartridge.



Educational Software Inc.
4565 Cherryvale Ave.
Soquel, CA 95073
(408) 476-4901 or
(800) 692-9520

Educational Software inc.